ORC 65-35
JUNE 1965

1

SOLUTION TECHNIQUES FOR THE TRAFFIC ASSIGNMENT PROBLEM

by
Stephen P. Bradley

OPERATIONS RESEARCH CENTER

COLLEGE OF ENGINEERING

UNIVERSITY OF CALIFORNIA - BERKELEY

24

SOLUTION TECHNIQUES FOR THE TRAFFIC ASSIGNMENT PROBLEM

by

Stephen P. Bradley
Operations Research Center
University of California, Berkeley

June 1965 ORC 65-35

This research has been partially sypported by the Army Research Office under Contract No. DA-31-124-ARO-D-331 and the Institute of Transportation and Traffic Engineering with the University of California, Berkeley. Reproduction in whole or in part is permitted for any purpose of the United States Government.

SOLUTION TECHNIQUES FOR THE TRAFFIC ASSIGNMENT PROBLEM

by

Stephen P. Bradley

1. INTRODUCTION

The ~~object of this~~ paper ~~is to~~ investigate the application of some large scale solution techniques to the so called traffic assignment problem, or multicommodity flow problem. As this is a network problem, there are two equivalent formulations. The first is based on the node-arc incidence matrix and is the usual approach. The second is based on an arc-chain matrix and the formulation is an extension of Ford & Fulkerson's multi-commodity work [5]. Decomposition techniques ~~will be~~ applied to the first formulation and a minimum chain network approach ~~will be~~ applied to the second. Then an extension of the arc-chain formulation ~~will~~ show it is equivalent to decomposition theory being applied to a slightly different node-arc formulation.

The great difficulty in handling the traffic assignment problem is the tremendous number of equations. If we are to consider a network with 1,000 arcs, 300 nodes, and 100 centroids (or commodities), where the centroids are a subset of the nodes through which all flow enters or leaves the network, we are faced with a linear program with 31,000 equations and 101,000 variables. Thus, techniques which are not principly network oriented must be discarded. Both compact triangularization [2] and generalized upper bounding [3] have been investigated and found wanting for the general problem, although these techniques may still prove to be useful in solving the restricted master.

2. NODE-ARC FORMULATION

The node-arc formulation of the traffic assignment problem has been extensively explained by Niels Jorgensen [6]. Therefore, it will be assumed that the motivation for the formulation is understood so that it will only be necessary to treat the problem from the mathematical standpoint. The statement of the problem is as follows:

$$\text{find} \quad f^{(k)} \geq 0 \;, \quad s \geq 0 \;, \quad \text{and min} \;\; Z$$

$$\text{subject to} \quad \begin{bmatrix} t & t & \dots & t & 0 \end{bmatrix} \qquad\qquad = \; Z$$

$$\begin{bmatrix} E & & & \\ & E & & \\ & & \ddots & \\ & & & E \\ I & I & \dots & I & I \end{bmatrix} \begin{bmatrix} f^{(1)} \\ f^{(2)} \\ \vdots \\ f^{(m)} \\ s \end{bmatrix} = \begin{bmatrix} q^{(1)} \\ q^{(2)} \\ \vdots \\ q^{(m)} \\ k \end{bmatrix} \qquad (2.1)$$

The kth commodity is defined as the flow that originates at the kth node. Thus, $Ef^{(k)} = q^{(k)}$ represents in matrix form the conservation equations for the kth commodity in terms of E, the node-arc incidence matrix, $f_i^{(k)}$, the flow of the kth commodity on arc A_i, and $q_i^{(k)}$, the exogenous flow of the kth commodity at node N_i. Note that $q_k^{(k)} > 0$ $q_i^{(k)} \leq 0$, for $i \neq k$. That is the kth commodity flows into the network at node N_k and out of the network at a specified subset of the remaining nodes. The last group of equations of (2.1) are the capacity constraints, ie. $\sum_k f_i^{(k)} + s_i = k_i$. This means that the total flow on arc A_i, $f_i = \sum_k f_i^{(k)}$, must be less than or equal to the capacity of the arc A_i. The objective is to minimize total travel time on the network, thus $\min Z = \sum_j t_j f_j$ or $\min Z = \sum_j \sum_k t_j f_j^{(k)}$. Note that

$$f_i^{(k)} \geq 0 \;, \; s_i \geq 0 \qquad\qquad \forall i, k$$

The technique to be employed is the general decomposition theory of Dantzig [1] with network techniques utilized on the subproblems. For simplicity only two subproblems will be considered, but the method obviously generalizes to any number of subproblems. Our problem with only two subproblems is thus:

$$\text{find } f^{(1)} \geqq 0 \text{ , } f^{(2)} \geqq 0 \text{ , } s \geqq 0 \text{ , and min } Z$$

$$\begin{array}{llll} \text{subject to} & tf^{(1)} + tf^{(2)} & = Z & \\ & Ef^{(1)} & = q^{(1)} & \qquad (2.1') \\ & \qquad Ef^{(2)} & = q^{(2)} & \\ & If^{(1)} + If^{(1)} + Is & = K & \end{array}$$

If we now consider the system of equations $Ef^{(k)} = q^{(k)}$, it is well known that any feasible solution of the system may be represented by an appropriate weighting of all the extreme point solutions of the system. Thus, we have for the two subproblems

$$\begin{array}{ll} f^{(1)} = \sum_1^K \lambda_j^{(1)} f_j^{(1)} & \left(\sum_1^K \lambda_j^{(1)} = 1 \; ; \; \lambda_i^{(1)} \geqq 0 \right) \\ f^{(2)} = \sum_1^L \lambda_j^{(2)} f_j^{(2)} & \left(\sum_1^L \lambda_j^{(2)} = 1 \; ; \; \lambda_i^{(2)} \geqq 0 \right) \end{array} \qquad (2.2)$$

Hence, letting $f_0 = -Z$, any solution of (2.1') may be represented as follows:

$$\begin{array}{ll} P_0 f_0 + \sum_1^K \lambda_j^{(1)} \left(Af_j^{(1)} \right) + \sum_1^L \lambda_j^{(2)} \left(Af_j^{(2)} \right) + BS = \begin{pmatrix} 0 \\ K \end{pmatrix} & \\ \sum_L^K \lambda_j^{(1)} \qquad\qquad\qquad\qquad = 1 & \qquad (2.3) \\ \qquad\qquad \sum_1^L \lambda_j^{(2)} \qquad\quad = 1 & \end{array}$$

Note that any solution $\lambda^{(1)}$, $\lambda^{(2)}$ to (2.3) determines an $f^{(1)}$, $f^{(2)}$ by equations (2.2). We denote the indicated linear transformations as

$$S_i^{(1)} = Af_i^{(1)} \; ; \; S_i^{(2)} = Af_i^{(2)} \qquad (2.4)$$

Thus, we have a problem that is equivalent to (2.1').

$$\text{find } \lambda^{(1)} \geq 0 \text{ , } \lambda^{(2)} \geq 0 \text{ and max } f_0$$

subject to

$$\begin{array}{lllll}
\pi: & P_0 f_0 + \sum_1^K S_j^{(1)} \lambda_j^{(1)} & + \sum_1^L S_j^{(2)} \lambda_j^{(2)} & + BS = \begin{pmatrix} 0 \\ K \end{pmatrix} & (2.5) \\
-\mu^{(1)}: & \sum_1^K \lambda_j^{(1)} & & = 1 & \\
-\mu^{(2)}: & & \sum_1^L \lambda_j^{(2)} & = 1 &
\end{array}$$

The multipliers associated with each equation are indicated at the left. Note that beginning with equation (2.3), the usual notation for the decomposition approach has been introduced. f_0 is simply equal to $-Z$ so that we maximize f_0 instead of minimizing Z. The indicated matrices A and B are simply:

$$P_0 = \begin{bmatrix} 1 \\ 0 \\ : \\ : \\ 0 \end{bmatrix} \qquad A = \begin{bmatrix} t_1 & t_2 & \dots & t_n \\ 1 & & & \\ & 1 & & \\ & & \ddots & \\ 0 & & & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 1 & & & \\ & 1 & & 0 \\ & & \ddots & \\ & 0 & & 1 \end{bmatrix}$$

The program (2.5) is refered to as the full master because it contains all the extreme point solutions of the subproblems. We will refer to a restricted master which is a subset of the full master that includes the current basis.

Assume for the moment that we have a basic feasible solution to (2.5). That is, we have a subset of k columns, of the K columns of subproblem one, and a subset of ℓ columns, of the L columns of subproblem two. Together with the f_0 column, these $k + \ell$ columns constitute a basis for the full master. If we then associate multipliers with each equation, we may determine their values in the usual manner:

$$\overset{0}{\pi} f_0 = 1 \; ; \; \overset{0}{\pi} S_i^{(1)} = \mu_0^{(1)} \; ; \; \overset{0}{\pi} S_i^{(2)} = \mu_0^{(2)}$$

Now that we have the multipliers $\overset{0}{\pi}$, $-\mu_0^{(1)}$, $-\mu_0^{(2)}$ we must check to see if we have an optimal solution. That is, we want to examine the columns of (2.5) that have the lowest cost. We must determine

$$\overset{0}{\pi} S_*^{(1)} = \min_i \overset{0}{\pi} S_i^{(1)} \; ; \; \overset{0}{\pi} S_*^{(2)} = \min_i \overset{0}{\pi} S_i^{(2)}$$

But we see that

$$\min \overset{0}{\pi} S_i^{(k)} = \min_i (\overset{0}{\pi} A) f_i^{(k)} = \min \overset{0}{\gamma} f^{(k)}$$
$$\text{subject to}$$
$$E f^{(k)} = q^{(k)} \; ; \; f^{(k)} \geq 0$$

which implies we must solve the subproblems of the form

$$\begin{aligned} &\text{find } f^{(k)} \geq 0 \text{ , min } Z^{(k)} \\ &\text{subject to } \overset{0}{\gamma} f^{(k)} = Z^{(k)} \\ &\qquad E f^{(k)} = q^{(k)} \\ &\qquad \text{where } \overset{0}{\gamma} = \overset{0}{\pi} A \end{aligned} \tag{2.6}$$

If the min $Z^{(k)} < \mu_0^{(k)}$, its relative cost factor is less than zero and is therefore a candidate to enter the basis. Since there are in general M subproblems, we should introduce a new column S_* according to the normal simplex rules, ie. S_* corresponding to $\min\limits_k [\overset{0}{\pi} S_*^{(k)} - \mu_0^{(k)}]$. If min $Z^{(k)} = \mu_0^{(k)}$ for all k , we have the optimal solution.

Consider now the subproblem that must be solved.

$$\gamma^0 f^{(k)} = Z^{(k)} \text{ (min)}$$
$$Ef^{(k)} = q^{(k)} \qquad (2.6')$$

$$\text{where } \gamma^0 = \pi^0 A = \pi^0 \begin{bmatrix} t_1 t_2 & \dots & t_n \\ 1 & & \\ & 1 & \\ & & 1 \\ & & & \ddots \\ & 0 & & & 1 \end{bmatrix} \text{ and } \pi^0 = 1$$

$$\text{which implies } \gamma^0 = \left\{t_i + \pi_i^0\right\}_{i=1}^{n} \triangleq \left\{c_i\right\}_{i=1}^{n}$$

Thus, our subproblems at each iteration are the same with a modified objective function. The subproblems are indeed merely minimal cost flow problems without any capacity constraints. These may easily be solved by network techniques. A straight-forward application of the out-of-kilter algorithm will yield a ready solution and has the advantage that the solution to the previous subproblem may be used as a starting solution to the next, thus speeding convergence. Another solution technique is the dynamic programming algorithm for the minimal spanning tree routed at the k^{th} node. With this method, it is also necessary to allocate flow according to the demands at each node and sum the resulting flows for each arc. Thus, we see that we have in general M uncapacitated minimal cost flow problems to solve in conjunction with a restricted master.

A few comments on the interpretation of the objective function are in order. Since $\pi_0^0 = 1$, the objective of the subproblems is

$$\gamma^0 f^{(k)} = Z^{(k)} \quad \text{where} \quad \gamma^0 = \left\{t_i + \pi_i^0\right\}_{i=1}^{n}$$

Therefore, each arc has a "cost" associated with it which is the sum of the given unconstrained travel time t_i , and an additional tarrif π_i^0 due to the demand for the arc exceeding its capacity. In effect, we put a price on the arcs with insufficient capacity to discourage flow from using these arcs. If the arc is not at capacity, we have more capacity than is demanded, and thus the price on the arc is zero. At each iteration the prices are re-evaluated and flow rerouted accordingly until the optimal is reached. Note that the objective functions are the same for all the subproblems. This is consistent with not discriminating between commodities on any arc. Thus, the price for any commodity to use a particular arc is the same for each.

The procedure is then to introduce the winning column into the restricted master, which consists of the current basis and the nonbasic slacks. We then optimize the restricted master, yielding a new basis. A new set of multipliers may then be calculated and the subproblems resolved with these new prices, etc. Since at each stage we are in effect optimizing the restricted master, the prices associated with the nonbasic slack variables will be $\pi_i^0 \geq 0$. Note the prices associated with basic slack variables are $\pi_i^0 = 0$.

The solution of the subproblems, although time consuming due to the large number, does not seem to be the restrictive aspect in this approach. Since these problems must be solved even in the uncapacitated traffic assignment problem, it is not surprising to find they must be solved in the capacitated case. At each iteration, ie. solution of the subproblems, we must find the new prices to add on to the arcs at capacity. Since we know that when slack occurs on arc A_i the corresponding price π_i^0 is equal to zero, we need only be concerned with the prices associated with the arcs at capacity and the multipliers $-\mu_0^{(k)}$ associated with the subsum equations. Thus, at least theoretically it is only necessary to invert a matrix of this rank. As the inversion process limits the size of the network problem we can handle, we would like to find an alternative method for finding the prices.

3. ARC-CHAIN FORMULATION

Instead of dealing with the usual node-arc formulation, it is well known that there is an equivalent formulation in terms of an arc-chain matrix. Since an interesting method of solution is available in this formulation, it will be presented here. First it is necessary to introduce some concepts relevant to the arc-chain formulation. Let $A = \left[a_{ij}\right]$ be an arc-chain matrix. Then we define

$$a_{ij} = \begin{cases} 1 & \text{if arc } A_i \text{ is in chain } C_j \\ 0 & \text{otherwise} \end{cases}$$

Now for the traffic assignment problem, we will change the definition of a commodity slightly. A commodity is defined to be flow between a particular origin-destination pair. Let $A^{(k)} = \left[a_{ij}^{(k)}\right]$ be the arc-chain matrix consisting of all chains connecting the kth origin-destination pair. In the previous formulation we had flow on an arc, here we have flow on a chain. Let $X_j^{(k)}$ be the flow of the kth commodity on chain C_j . We may now represent the capacity constraint on the ith arc as follows:

$$\sum_{j_1} a_{ij}^{(1)} X_j^{(1)} + \sum_{j_2} a_{ij}^{(2)} X_j^{(2)} + \ldots + \sum_{j_m} a_{ij}^{(m)} X_j^{(m)} + s_i = K_i \tag{3.1}$$

$$i = 1,2, \ldots, n$$

This merely means that the flow in arc A_i is equal to the sum of the chain flows through the arc A_i , and this total flow plus the slack must equal the capacity of the arc A_i .

The second requirement of the problem is that there is a specified amount of flow between any origin-destination pair. Therefore, the total chain flows associated with the kth origin-destination pair must equal the required flow,

$$\sum_{j_k} X_{j_k}^{(k)} = V^{(k)} \qquad \forall k \tag{3.2}$$

Of course, all the chain flows must be nonnegative, $X_j^{(k)} \geq 0$.

Finally, we define $c_j^{(k)}$ as the total travel time on chain C_j between the kth origin-destination pair. Since we wish to minimize total travel time on the network, our objective function is

$$\min \ Z = \sum_{j_1} c_j^{(1)} X_j^{(1)} + \sum_{j_2} c_j^{(2)} X_j^{(2)} + \ldots . + \sum_{j_m} c_j^{(m)} X_j^{(m)} \qquad (3.3)$$

Thus, we have formulated the traffic assignment as the following linear program.

$$\min \sum_k \sum_j c_j^{(k)} X_j^{(k)}$$

subject to

$$\begin{array}{lllll}
\pi_i: & \sum_{j_1} a_{ij}^{(1)} X_j^{(1)} + \sum_{j_2} a_{ij}^{(2)} X_j^{(2)} + \ldots . + \sum_{j_m} a_{ij}^{(m)} X_j^{(m)} + s_i & = k_i & \forall i & \\
-\mu^{(1)}: & \sum_{j_1} X_j^{(1)} & = V^{(1)} & & \\
-\mu^{(2)}: & \quad \sum_{j_2} X_{j_2}^{(2)} & = V^{(2)} & & (3.4) \\
\vdots & \qquad \ddots & \vdots & & \\
-\mu^{(m)}: & \qquad\qquad \sum_{j_m} X_j^{(m)} & = V^{(m)} & &
\end{array}$$

An extension of the multi-commodity work of Ford & Fulkerson [5] leads to a solution of the problem. Assume for the moment we have a known basic feasible solution to (3.4). Thus, we can calculate $\overset{0}{\pi}_i$, $-\mu_0^{(k)}$ as we did in the previous section. Note that these prices are the same type as those used in the decomposition approach, thus their sign is opposite that of the usual simplex multiplier. Let us now consider whether the prices are optimal for the program. Pricing out we have

$$\bar{c}_j^{(k)} = c_j^{(k)} + \sum_i \overset{0}{\pi}_i a_{ij}^{(k)} - \mu_0^{(k)} \geq 0 \qquad \forall j , k \qquad (3.5)$$

for the optimality criterion. Notice that to test optimality it is not necessary to look at all $\bar{c}_j^{(k)}$. If we look at $\bar{c}_s = \min_{k,j} \bar{c}_j^{(k)}$, we know if we are optimal by whether $\bar{c}_s$ is nonnegative or not; if $\bar{c}_s \geq 0$, we are optimal. Also note that $c_j^{(k)}$, the total travel time on chain C_j between the kth origin-destination pair may be represented as follows:

$$c_j^{(k)} = \sum_i t_i a_{ij}^{(k)}$$

where t_i is the travel time on arc A_i just as before. The total travel time on a chain equals the sum of the travel times on the arcs in that chain. Thus, we may write

$$\bar{c}_j^{(k)} = \sum_i t_i a_{ij}^{(k)} + \sum_i \overset{0}{\pi}_i a_{ij}^{(k)} - \mu_0^{(k)}$$

or (3.6)

$$\bar{c}_j^{(k)} = \sum_i (t_i + \overset{0}{\pi}_i) a_{ij}^{(k)} - \mu_0^{(k)}$$

$t_i \geq 0$ and we will show $\overset{0}{\pi}_i \geq 0$, thus $(t_i + \overset{0}{\pi}_i) \geq 0$. If we now consider $(t_i + \overset{0}{\pi}_i)$ as the length of the arcs, we may interpret the minimum relative cost factor in an interesting way.

$$\bar{c}_s^{(k)} = \left[\min_j \sum_i (t_i + \overset{0}{\pi}_i) a_{ij}^{(k)}\right] - \mu_0^{(k)} \qquad \forall k$$

$$\bar{c}_s = \min_k \bar{c}_s^{(k)} \tag{3.7}$$

Since $\mu_0{}^{(k)}$ is a constant for a particular origin-destination pair, we need only look at

$$\min_j \sum_i (t_i + \overset{0}{\pi}_i) a_{ij}^{(k)}$$

We see that this merely represents the shortest distance between the k th origin-destination pair, when $(t_i + \overset{0}{\pi}_i)$ is considered as the length of the arc. Therefore to check optimality, we need only find the shortest paths associated with the origin-destination pairs, subtract the appropriate $\mu_0^{(k)}$ and take the minimum over all k .

$$\bar{c}_s = \min_k \left\{ \min_j \left[\sum_i (t_i + \overset{0}{\pi}_i) a_{ij}^{(k)} \right] - \mu_0^{(k)} \right\} \tag{3.8}$$

Thus, we see that is is not necessary to explicitly enumerate all the chains of the network. Only the shortest chain for each commodity must be examined. Not having to know the arc-chain matrix explicitly is absolutely necessary for this method to be computationally feasible.

A few comments on the signs of the multipliers are now in order. We have defined the multipliers consistent with the decomposition theory, and thus they are opposite in sign with respect to the usual simplex multipliers. The multipliers associated with the current basis of the restricted master are defined by

$$\overset{0}{\pi} P_j = 0 \quad j \neq 0 \ ; \quad \pi \cdot P_0 = 1 \qquad \text{for basic columns}$$

It is easy to see that if s_i is basic $\overset{0}{\pi}_i = 0$. Now if s_i is nonbasic $\overset{0}{\pi}_i \geq 0$ since we are at each stage optimal with respect to a subset of columns including the current basis and the nonbasic slack. Note this is exactly analogous to the decomposition situation. The $\overset{0}{\pi}_i$ are in fact prices, just as they were before, and represent additional tarrifs associated with the use of the capacited arcs. As before, this additional tarrif is added to the unconstrained travel time t_i forming a new "cost", and the flow is rerouted accordingly. We may also determine the sign of $-\mu_0^{(k)}$ in an easy manner. Since at least one variable from each commodity subset must be basic [3], we may write

$$-\mu_0^{(k)} + \sum_i (t_i + \overset{0}{\pi}_i) a_{ij}^{(k)} = 0 \qquad \text{for some } j$$

or rewriting and noting that $t_i \geq 0$ and $\overset{0}{\pi_i} \geq 0$, we have

$$-\mu_0^{(k)} = \sum(t_i + \overset{0}{\pi_i})a_{ij}^{(k)} \geq 0 \qquad \forall k \qquad (3.9)$$

$\mu_0^{(k)}$ is the length of the shortest path between the kth origin-destination pair that is currently in the basis. All such paths for the kth commodity must have the same length.

Rather than discussing the relative merits of this arc-chain formulation at this point, we will put this aside until the end of the next section. In the next section, we will show that the arc-chain formulation is equivalent to a decomposition formulation and thus the problems encountered will be similar.

4. EXTENSION OF THE ARC-CHAIN FORMULATION

The similarity between the two previous formulations lead us to investigate further their relationship. The outstanding similarity is that in each case prices are put on the arcs at capacity which are added directly to the uncapacitated travel times of the arcs. The flow allocation is then re-examined with respect to these new composite travel times. Also in the decomposition approach we solve a subproblem which is a minimal spanning tree, or shortest route to all nodes of the network. In the arc-chain formulation, we solve a shortest route between an origin-destination pair. Let us then make the following transformation of variables on the arc-chain formulation (3.4):

$$X_j^{(k)} = \lambda_j^{(k)} V^{(k)} \qquad \forall k \tag{4.1}$$

This results from dividing each flow requirement equation by the required flow; in effect normalizing the equations. Thus, we may write the arc-chain problem as follows in terms of the new variables.

$$\min \quad V^{(1)} \sum_{j_1} C_j^{(1)} \lambda_j^{(1)} + V^{(2)} \sum_{j_2} C_j^{(2)} \lambda_j^{(2)} + \dots + V^{(m)} \sum_{j_m} C_j^{(m)} \lambda_j^{(m)}$$

subject to

$$\begin{aligned}
V^{(1)} \sum_{j_1} a_{ij}^{(1)} \lambda_j^{(1)} + V^{(2)} \sum_{j_2} a_{ij}^{(2)} \lambda_j^{(2)} + \dots + V^{(m)} \sum_{j_m} a_{ij}^{(m)} \lambda_j^{(m)} + s_i &= k_i \qquad \forall i \\
\sum_{j_1} \lambda_j^{(1)} &= 1 \\
\sum_{j_2} \lambda_j^{(2)} \quad \ddots \quad &= 1 \\
&\vdots \\
\sum_{j_m} \lambda_j^{(m)} &= 1
\end{aligned} \tag{4.2}$$

Now our program looks exactly like the restricted master of a decomposition program. Note that each subsum of $\lambda_j^{(k)}$ equals one and thus the $\lambda_j^{(k)}$ may be considered as weights associated with the corresponding columns. Also, it is easy to see that each column is a path between the k^th origin-destination pair with the appropriate flow on it, namely $V^{(k)}$. Thus, equation set (4.2) is really a full master for a decomposition problem whose subproblems are shortest route problems between an origin-destination pair. By full master, we mean that (4.2) contains all the extreme point solutions of the subproblems. The reason why we do not have to explicitly know the arc-chain matrix is simply that we bring up the needed extreme point solution, ie. shortest path, from the subproblem only when it prices out negative. Since network techniques are used on the subproblems, they could be formulated in the usual node-arc manner. Thus, we have shown that the arc-chain formulation and solution technique is equivalent to the decomposition technique. The decomposition problem for which (4.2) is the full master is as follows:

$$\text{find } f^{(k)} \geqq 0 \quad s \geqq 0 \text{ , and min } Z$$

subject to

$$\begin{bmatrix} t & t & t & \cdots & t & 0 \end{bmatrix} = Z \quad (\text{min})$$

$$\begin{bmatrix} E & & & & & \\ & E & & & & \\ & & E & & & \\ & & & \ddots & & \\ & & & & E & \\ I & I & I & \cdots & I & I \end{bmatrix} \begin{bmatrix} f^{(1)} \\ f^{(2)} \\ f^{(3)} \\ \vdots \\ f^{(P)} \\ s \end{bmatrix} = \begin{bmatrix} q^{(1)} \\ q^{(2)} \\ q^{(3)} \\ \vdots \\ q^{(P)} \\ K \end{bmatrix} \quad \text{where} \quad q^{(k)} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -v^{(k)} \\ 0 \\ \vdots \\ 0 \\ v^{(k)} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (4.3)$$

The similarity between this and our original node-arc formulation is very great.

Where before we had M usbproblems, which involved finding the shortest path to all other nodes, here we have M(M-1) subproblems, which involve finding the shortest path between an origin-destination pair.

Thus, we have shown that the arc-chain formulation simplifies the subproblem considerably, but at the same time it expands the number of subproblems considerably. Both the node-arc and the arc-chain procedures depend on the same principles. We solve a number of routing problems given a set of prices on the capacited arcs and check optimality. If we are not optimal, we re-evaluate our prices and resolve our routing problems again with respect to the new prices. In both procedures the prices associated with the arcs not at capacity are zero. Thus, in the arc-chain formulation we must theoretically invert a matrix the rank of the number of arcs at capacity plus the number of requirement equations. A criticism that might be lodged against the arc-chain formulation is that the number of subproblems has become prohibitively large. Indeed the number of subproblems places a bound on the size problem we can handle, but in fact these M(M-1) shortest route problems may be solved more quickly than the M unconstrained minimal cost flow problems. This is true since the M(M-1) problems are solved by applying the dynamic programming algorithm for minimal spanning tree routed at a particular node M times. Each application solves M-1 of the subproblems. Since this must be done in the minimal cost flow situation plus assigning flows and adding appropriately for each arc, it will be quicker. The difficulty with the method is that there are considerably more multipliers associated with subsum equations to be calculated.

Note that in each case we assumed that we had a basic feasible solution to the restricted master. This is usually not the case, but the method outlined by Dantzig [1] may be used to obtain one. We simply add artificial variables to the restricted master and minimize the infeasibility form. The method is the same as that described for the decomposition approach. Here the unconstrained travel time on the arcs is considered zero, and a price is put on those arcs that are associated with an infeasible equation.

Thus, we have shown how a pricing mechanism forces the re-allocation of commodity flow until an overall optimum is attained. For a given set of prices, we check to see if the current flow allocation is optimal; and if not we find new prices for the arcs and reroute flow accordingly. The simplex criteria tell us which new route (chain) should be considered and which old route should be dropped. Then new prices are found consistent with the new basis and the processes repeats itself.

5. COMPUTATIONAL STATUS

Now that the theory has been demonstrated, it seems appropriate to say a few more words concerning computational possibilities. We have indeed shown that the problem can be formulated as two slightly different decomposition problems. The first defines commodity as that flow leaving the k^{th} node. Thus, for M commodities, or centroids, we have M uncontrained minimal cost flow problems. The second, an extension of Ford & Fulkerson's work, defines commodity as flow between the k^{th} origin-destination pair. Thus, for M centroids we have $M(M-1)$ shortest route problems. The differences are only slight and only slightly different solution techniques are implied.

The unconstrained minimal cost flow problems of the first formulation may be solved in two ways. First, a straight forward application of the out-of-kilter algorithm will yield a rapid solution as the problems are unconstrained. Also, since the objective function is the same for each subproblem at any iteration, the solution of the i^{th} subproblem can be used as a starting solution for the $i + 1^{th}$ subproblem at any iteration - thus speeding convergence. As an alternative, it appears that using the solution of the i^{th} subproblem at iteration t as the starting solution for the i^{th} subproblem at iteration $t + 1$ will make convergence extremely rapid as the right hand sides will be identical and the objective functions only slightly modified. Of course, this entails storing in some manner the M solution vectors of the subproblems from iteration to iteration. Note that this causes no real difficulty as the solution vectors currently in the basis of the restricted master will be, if not equivalent, just as useful. The second method to solve the subproblems, which has been used in practice, is to solve a minimum tree routed at a particular node for each subproblem. This is done extremely rapidly by application of the dynamic programming algorithm for "shortest path" (actually shortest path to all nodes and thus minimum tree). This computation may be shortened by judicious choice of the order in which the subproblems are solved. That is, if the appropriate node is chosen adjacent to the one for which the minimum tree was just calculated, a great deal of the new minimum tree will be the same. Unfortunately, their presently exists no systematic technique for doing this. To find the column to introduce into the basis, we need the flows on the arcs. Thus, for each destination node of a subproblem, we traverse the shortest route to the origin (ie. in the reverse direction) assigning the demand $q_i^{(k)}$ on each arc A_i. When this is done for each destination node, we simply sum the resulting demands on each arc to obtain the required total flow on that arc. Although this method of solution is more difficult to describe than the out-of-kilter method, many argue that it is as efficient.

Now with the subproblems solved at some iteration, we want to introduce the appropriate column into the restricted master. We could use the straight forward application of a linear programming code to optimize the restricted master, as this yields the desired prices directly. Clearly, if this method is used all those columns that currently price out negatively from solution of the subproblems should be introduced into the restricted master; then the restricted master is optimized. The number of equations is then equal to the number of arcs plus the number of commodities (subproblems). Thus, the rank of the restricted master constrains the size problem you can solve (ie. as large as the linear programming code employed). There is an alternative approach, although it has not yet been fully worked out. That is, we utilize the fact we know the price is zero on all arcs not at capacity. Thus, if we select our matrix to invert, it must only be of rank equal to the number of capacitated arcs plus the number of commodities. This is a sizable reduction as the number of capacitated arcs is rarely more than sixty percent of the total and often as low as thirty percent. This approach has the obvious difficulty that the number of capacitated arcs may change from iteration to iteration.

The second formulation is of interest because the technique for the subproblems is straight forward. Here we have M(M-1) shortest route problems between origin-destination pairs. We simply apply the dynamic programming algorithm M times; each application yielding the solutions to M-1 of the shortest route subproblems. Note that no back search is necessary to find the flows. The only difficulty with this solution approach in general is that the restricted master has n equations for the capacity constraints as before but also M(M-1) commodity equations instead of M. Therefore, the number of commodity equations may become prohibitive unless special techniques are employed. For a large problem, it is suggested that the generalized upper bounding algorithm of Dantzig and Van Slyke be utilized [3]. The algorithm given in Vol. II of their paper has a "working basis" of rank equal to the number of arcs.

In conclusion, it would appear that a reasonable size problem can be solved straight forwardly by either method. The bound on this "reasonable size" is placed either by the linear programming code size utilized to solve the restricted master or by the computing time utilized in solution of the subproblems. For present computers, a chain job configuration would have to be used on a large problem; this could be quite time consuming, but future machines will eliminate this difficulty.

At this stage there are three areas where further research seems justified. First, an approach using only network techniques would be desirable, but the simple pricing mechanism implied by the decomposition approach would seem hard to improve upon. Second, an efficient technique to calculate the prices that uses only the information associated with the capacitated arcs would greatly increase the size problem the above techniques could accomodate. Third, in practice a systematic technique of guessing the prices might be the best. Finally, as computer technology increases the amount of rapid access storage available to the user, the above techniques should prove to be extremely rapid.

APPENDIX

An intersting extension of the classical traffic assignment problem has been proposed by A. Ridley.† The basic concern is to add a budget which can be used to increase the capacity of any arc. Thus, the capacity constraints for the problem take on a different form. The right hand side now becomes the sum of two capacities, the present existing capacity plus that capacity which can be added by spending money on that particular arc. Thus, we have

$$\sum_k f_j^{(k)} \leq K_j + k_j \quad \text{on} \quad \sum_k f_j^{(k)} + s_j = K_j + k_j \qquad \forall j \tag{A.1}$$

Since we want to minimize total travel time on the network, we could increase the capacities on all the shortest paths; but this is not possible with only a finite amount of money. If we call B , the budget or the total amount we can spend; and call r_i the cost per unit increase in capacity on the ith arc we have

$$\sum_j r_j k_j \leq B \quad \text{on} \quad \sum_j r_j k_j + s' \tag{A.2}$$

That is the amount of money spent on each arc, summed over all arcs, must be less than or equal the total budget. The new problem in matrix notation is then

$$\text{find } f^{(k)} \geq 0 \quad s \geq 0 \quad k \geq 0 \quad s' \geq 0 \quad \min Z$$

$$\text{subject to} \quad \begin{bmatrix} E & & & & & & \\ & E & & & & & \\ & & \ddots & & & & \\ & & & E & & & \\ I & I & \dots & I & I & -I & 0 \\ 0 & 0 & \dots & 0 & 0 & r & 1 \end{bmatrix} \begin{bmatrix} f^{(1)} \\ f^{(2)} \\ \vdots \\ f^{(m)} \\ s \\ k \\ s' \end{bmatrix} = \begin{bmatrix} q^{(1)} \\ q^{(2)} \\ \vdots \\ q^{(m)} \\ K \\ B \end{bmatrix}$$

$$\begin{bmatrix} c & c & \dots & c & 0 & 0 & 0 \end{bmatrix} = Z(\min)$$

†Private communication

The reason for considering this variation is to show that the proposed solution technique also solves this problem as an easy extension. The subproblems are obviously exactly the same as they were in the initial formulation. Thus, let us now consider the restricted master.

$$\text{find } \lambda^{(1)} \geq 0 \quad \lambda^{(2)} \geq 0 \quad s \geq 0 \quad k \geq 0 \quad s' \geq 0 \quad \max \ f_0$$

$$P_0 f_0 + \sum_1^k S_i^{(1)} \lambda_i^{(1)} + \sum_1^L S_i^{(2)} \lambda_i^{(2)} + Is - Ik = K \tag{A.4}$$

$$rk + s' = B$$

$$\sum_1^k \lambda_i^{(1)} = 1$$

$$\sum_1^L \lambda_i^{(2)} = 1$$

Note that this only adds one more equation to the problem whil· adding several more variables. Thus, any straight forward method for optimising the restricted master, such as the revised simplex method, will yield prices just as it did before. In fact, the solution technique for this problem is identical with that of the first formulation except shortcuts for obtaining prices may not be applicable.

Since the objective is to minimize total travel time regardless of cost, assuming we are within the budget, the solution is intuitive. We will spend money to increase the capacity of the shortest paths on critical arcs until one of two things happens. Either all commodities will be traveling over their shortest paths only or the budget will be constraining.

BIBLIOGRAPHY

[1] Dantzig, George B., LINEAR PROGRAMMING AND EXTENSIONS, Princeton University Press, Princeton, New Jersey, (1963).

[2] Dantzig, George B., "Compact Basis Triangularization for the Simplex Method," Operations Research Center Report ORC 62-28, University of California, Berkeley.

[3] Dantzig, George B., and Richard M. Van Slyke, "Generalized Upper Bounded Techniques for Linear Programming - I & II," Operations Research Center Report ORC 64-17, ORC 64-18, University of California, Berkeley.

[4] Ford, L. R., and D. R. Fulkerson, "A Suggested Computation for Maximal Multi-Commodity Network Flows" Management Science, (October 1958).

[5] Ford, L. R., and D. R. Fulkerson, FLOWS IN NETWORKS, Princeton University Press, Princeton, New Jersey, (1962).

[6] Jorgensen, Niels O., "Some Aspects of the Urban Traffic Assignment Problem," ITTE Report, University of California, Berkeley.

[7] Mosher, "A Capacity Restraint Algorithm for Assignment of Flow to a Transport Network," ITTE Report, University of California, Los Angeles.